# Spatial Birth-Death-Swap Chains

**Mark L. Huber**
*Department of Mathematics and Computer Science, Claremont McKenna College*
*mhuber@cmc.edu*

**Abstract**

Markov chains have long been used for generating random variates from spatial point processes. Broadly speaking, these chains fall into two categories: Metropolis-Hastings type chains running in discrete time and spatial birth death chains running in continuous time. These birth death chains only allow for removal of a point or addition of a point. In this work it is shown that the addition of transitions where a point is moved from one location to the other can aid in shortening the mixing time of the chain. Here the mixing time of the chain is analyzed through coupling, and use of the swap moves allows for analysis of a broader class of chains. Furthermore, these swap moves can be employed in perfect sampling algorithms via the dominated Coupling from the Past procedure of Kendall and Møller. This method can be applied to any pairwise interaction model with repulsion. In particular, an application to the Strauss process is developed in detail, and the swap chains are shown to be much faster than standard birth death chains.

**Keywords:** Perfect simulation, coupling from the past, swap moves, birth death process, spatial point processes, Strauss process

MSC-class: 60G55 (primary); 60J22 (secondary)

## 1 Introduction

Spatial point processes are in wide use in statistical modeling (see [16] for an overview). It is typical to model finite point processes as being absolutely continuous with respect to a Poisson point process. That is, they have a density $f(x)/c$ where $f(x)$ is an easily computable function but the normalizing constant $c$ of the density is impractical to compute exactly. A Monte Carlo algorithm gains information about the density by studying random variates drawn from the distribution the density describes.

In order to obtain such variates, a Markov chain is constructed whose stationary distribution matches the target distribution. Metropolis-Hastings chains for this problem run in discrete time (see [10]), and the spatial birth death chain approach of Preston [23] runs in continuous time. In [6] it was noted that for some problems the Metropolis-Hastings approach appears faster than the continuous time methods.

There is a drawback to all of these Markov chain Monte Carlo methods in that unless the mixing time of the Markov chain is known ahead of time, the quality of the

variates obtained is suspect. While heuristics such as the autocorrelation test (see [9]) can provide proof that a chain has not mixed, it is often difficult to establish the positive claim that a chain has mixed.

Perfect simulation algorithms are a solution to this problem, they generate samples exactly from the desired distribution without the need to know the mixing time of a Markov chain. Kendall [19] showed how the Coupling from the Past (CFTP) idea of Propp and Wilson [24] could be used together with a spatial birth and death chain to obtain samples from area-interaction processes. Kendall and Møller [20] showed how this method could be extended to any locally stable point process using a method they called *dominated CFTP*. They also considered perfect sampling Metropolis-Hastings chains, but restricted these chains to only adding or deleting a point at each step.

So while [6] indicates that Metropolis-Hastings chains can beat continuous time spatial birth death chains, [20] shows how CFTP can be used with the latter chains. In this work a new swap move is introduced that speeds up convergence, while still allowing dominated CFTP to be used for perfect simulation.

In Section 2.1 the theory behind spatial birth-death chains with the new swap move is developed. Section 3 describes the framework for how these chains are employed for locally stable processes, and uses these moves for several examples. Section 4 then reviews the use of dominated coupling from the past for these types of chains, and shows how the addition of swap moves fits into this protocol. In Section 5, the examples of Section 3 are simulated, both with and without the swap moves. Section 6 bounds the expected running time of the procedure for a restricted class of models.

## 2 Spatial point processes

Dyer and Greenhill [7] first introduced a swap move for hard core processes living in discrete spaces. Here their method is extended to more general point processes where the points can lie in continuous spaces and repulsion can be weaker than hard core.

Following [4], the 'Carter-Prenter exponential space' will be used as the state space. Consider a separable measure space $(S, \mathcal{B}, \lambda)$ where $0 < \lambda(S) < \infty$ and $\mathcal{B}$ contains all singletons. For example, both finite spatial point processes and marked point processes can be included in this framework.

For simple spatial data $S$ is usually a bounded Borel set of $\mathbf{R}^2$ and $\lambda$ is typically Lebesgue measure. Other information can then be added to the points as marks. For example, Harkness and Isham [13] studied locations of ants nests in a rectangular region $R$. Since there were two types of ants, $S = R \times \{0, 1\}$ and $\lambda$ is the cross product of Lebesgue measure and a measure on $\{0, 1\}$.

As usual for continuous applications, suppose $(\forall s \in S)(\lambda(\{s\}) = 0)$. Now consider generating $N$ as a Poisson random variable with parameter $\lambda(S)$. Then with $N$ in hand, draw $X_1, \ldots, X_N$ iid from $\lambda(\cdot)/\lambda(S)$. Then $\{X_1, \ldots, X_N\}$ (called a *configuration*) is a draw from a Poisson point process with intensity measure $\lambda(\cdot)$ over $S$. Let $\mu$ be the distribution of $\{X_1, \ldots, X_N\}$, and $\Omega$ the set of all possible configurations. See [23] for more details and precise definitions of $\mu$ and $\Omega$.

Now consider point processes that are absolutely continuous with respect to $\mu$ with

density $f$ satisfying a *local stability condition* (as in [20]):

$$(\exists K > 0)(\forall x \in \Omega)(\forall v \in S)(f(x+v) \leq K f(x)), \tag{1}$$

where $x+v$ denotes the configuration $x \cup \{v\}$. Many point processes of interest meet this condition, including the area-interaction process [2, 27], the Strauss process [18, 26], and the continuous-random cluster model [21, 12].

### 2.1 Spatial birth death swap chains

The presentation of the swap move given here follows the framework of Preston [23], who introduced the use of continuous time spatial birth death chains for these problems. These chains are examples of *jump processes*, where at a given state $x$, the chain stays in the state for an exponential length of time with expected value given by $1/\alpha(x)$. The state then jumps to a new state using kernel $\mathbf{K}$, so the probability the new state is in $A$ is $\mathbf{K}(x, A)$) independent of the past history (see [8], Chapter X for the details of jump processes.)

In the Preston framework, the rate of births (adding points to the configuration) and deaths (deleting points from the configuration) depends only on the current state:

- There exists a nonnegative measurable birth rate function $b$ from $\Omega \times S$ equipped with the standard product $\sigma$-field to $\mathbf{R}$ with the Borel $\sigma$-field.
- There exists a nonnegative measurable death rate function $d$ from $\Omega \times S$ equipped with the standard product $\sigma$-field to $\mathbf{R}$ with the Borel $\sigma$-field. Furthermore, $w \in x \Rightarrow d(x, w) > 0$ and $w \notin x \Rightarrow d(x, w) = 0$.

To this birth death framework we now add a swap rate:

- There exists a nonnegative measurable swap rate function $s$ from $\Omega \times S \times S$ equipped with the standard product $\sigma$-field to $\mathbf{R}$ with the Borel $\sigma$-field. Furthermore, $w \notin x \Rightarrow s(x, w, v) = 0$.

Here $b(x, v)$ is the rate at which points $v$ are added to $x$, $d(x, w)$ is the rate at which point $w$ is removed from $x$, and $s(x, w, v)$ is the rate at which point $w$ is removed and point $v$ is added to $x$. In order to make the construction precise, from $b, d$ and $s$ the rate of change $\alpha(x)$ and kernel $\mathbf{K}$ for the Markov chain must be defined.

For all $A \in \mathcal{B}$, let $K_b(x, A) = \int_{v \in S} b(x, v)\mathbf{1}(x + v \in A)\ \lambda(\mathrm{d}v)$. When $K_b(x, \Omega) < \infty$ for all $x$ in $\Omega$, a birth kernel can be defined:

$$\mathbf{K}_b(x, A) = K_b(x, A)/K_b(x, \Omega), \tag{2}$$

with rate $r_b(x) = \int_{v \in S} b(x, v)\ \lambda(\mathrm{d}v)$.

Similarly, $K_d(x, A) = \sum_{w \in x} d(x, w)\mathbf{1}(x - w \in A)$ which always has a finite number of terms and so

$$\mathbf{K}_d(x, A) = K_d(x, A)/K_d(x, \Omega), \tag{3}$$

with rate $r_d(x) = \sum_{v \in x} d(x, v)$.

Lastly, set $K_s(x, A) = \sum_{w \in x} \int_{v \in S} s(x, w, v)\mathbf{1}(x - w + v \in A)\ \lambda(\mathrm{d}v)$. When $K_s(x, \Omega) < \infty$ for all $x \in \Omega$, let

$$\mathbf{K}_s(x, A) = K_s(x, A)/K_s(x, \Omega), \tag{4}$$

and $r_s(x) = \sum_{w \in x} \int_{v \in S} s(x, w, v)\ \lambda(\mathrm{d}v)$. This makes the overall rate

$$\alpha(x) = r_b(x) + r_d(x) + r_s(x), \tag{5}$$

and the overall kernel:

$$\mathbf{K}(x, A) = \mathbf{K}_b(x, A)\frac{r_b(x)}{\alpha(x)} + \mathbf{K}_d(x, A)\frac{r_d(x)}{\alpha(x)} + +\mathbf{K}_s(x, A)\frac{r_s(x)}{\alpha(x)}. \tag{6}$$

Harris recurrence guarantees that a Markov process has a unique invariant measure (see [1] for details of Harris recurrence in continuous time). Kaspi and Mandelbaum [17] showed that a continuous time chain is Harris recurrent if and only if there exists a nonzero $\sigma$-finite measure where the chain almost surely hits sets with positive measure.

In particular, for all the chains here, the death rate equals the number of points in the configuration, and the birth rate is bounded above by a constant. This forces the chain to visit the empty configuration infinitely often, making it Harris recurrent.

It is well-known (see Proposition 8.1 of [23]) that for jump processes with a unique invariant distribution, the density $f$ is invariant if and only if for all $A \in \mathcal{F}$:

$$\int_A \alpha(z) f(z)\ \mathrm{d}\mu(z) = \int_\Omega \mathbf{K}(y, A)\alpha(y) f(y)\ \mathrm{d}\mu(y), \tag{7}$$

for $\alpha$ as in (5). Detailed balance conditions for time-reversibility of jump processes are:

$$f(x)\alpha(x)\mathbf{K}(x, \mathrm{d}y)\ \mathrm{d}\mu(x) = f(y)\alpha(y)\mathbf{K}(y, \mathrm{d}x)\ \mathrm{d}\mu(y). \tag{8}$$

For moves from configurations with $n$ points to those with $n + 1$ (or vice versa), the detailed balance conditions are satisfied [23, 25] when the rate of births balance the rate of deaths with respect to $f$. So

$$f(x)b(x, v) = f(x + v)d(x + v, v). \tag{9}$$

Swap moves stay inside the same dimensional space, and it is straightforward to show that reversibility for swap moves holds when

$$f(x)s(x, w, v) = f(x + v - w)s(x + v - w, v, w). \tag{10}$$

## 3 Applications

The description of the continuous time birth death swap chain given here mirrors that of Kendall and Møller [20] for Preston's continuous time birth death chains. In Section 4 their method of dominated coupling from the past (dCFTP) will be used with these chains to obtain a perfect simulation algorithm. Consider a locally stable point process satisfying (1) with density $f$.

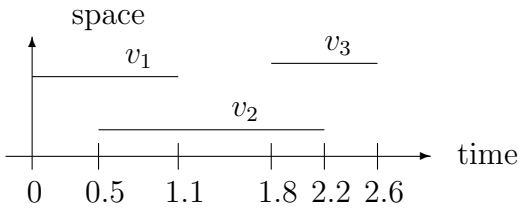
**Dominating process** To put the framework of Preston into practice, let the *dominating process* $(D, M)$ be a marked spatial point process over times $t \in (-\infty, \infty)$, where $D$ is a spatial birth death process with constant birth rate $K$ and constant death rate 1 over $S$. This makes the total birth rate equal to $K\lambda(S)$, and the total death rate equal to the number of points in the configuration. The process $M$ contains an extra mark for each point in the process that is a uniform draw on $[0, 1]$ that is independent of other marks, the point location, and the location of all previous points in the process.

The following pseudocode generates the Poisson point process $D(0)$.

**Start dominating process**
*Input:* $K$, $\lambda(\cdot)$
*Output:* $D(0)$
**Draw** $N_0 \leftarrow \text{Pois}(K\lambda(S))$
**Let** $D(0) \leftarrow \emptyset$
**For** $i$ from 1 to $N_0$ do
  **Draw** $v \leftarrow \lambda(\cdot)/\lambda(S)$
  **Let** $D(0) \leftarrow D(0) + v$

With the starting state, it is now possible to generate moves: births or deaths of points in the dominating chain. $\text{Exp}(1/r)$ denotes an exponential random variable with mean $r$, and $\text{Exp}(0) = \infty$ with probability 1. $\text{Unif}(x)$ denotes a uniform draw from the points in the configuration $x$.

**Dominating process**
*Input:* $x$ (starting configuration), $n$ (number of events)
*Output:* $x$ (configuration after $n$ events), $A$ (list of births and deaths)
**Let** $t \leftarrow 0$
**For** $i$ from 1 to $n$ do
  **Draw** $t_b \leftarrow \text{Exp}(K\lambda(S))$, **draw** $t_d \leftarrow \text{Exp}(\#x)$, **draw** $M \leftarrow \text{Unif}([0, 1])$
  **If** $t_b < t_d$ (then there is a birth)
    **Draw** $v \leftarrow \lambda(\cdot)/\lambda(S)$, **let** $x \leftarrow x + v$
    **Add** row $[t$ "birth" $v$ $M]$ to $A$
  **Else** (there is a death)
    **Draw** $w \leftarrow \text{Unif}(x)$, **let** $x \leftarrow x - w$
    **Add** row $[t$ "death" $w]$ to $A$
  **Let** $t \leftarrow t + \min\{t_b, t_d\}$




$A$

| time | type | pt. | mark |
|---|---|---|---|
| 0.5 | birth | $v_2$ | .9501 |
| 1.1 | death | $v_1$ | |
| 1.8 | birth | $v_3$ | .6068 |
| 2.2 | death | $v_2$ | |
| 2.6 | death | $v_3$ | |

Figure 1: Sample run of Dominating process

With the dominating process in hand, it is possible to build any locally stable process $X$ with stationary density $f$ that satisfies $X(t) \subseteq D(t)$ for all $t$ as follows. Suppose $X(T) \subseteq D(T)$ for a particular time $T$. Now consider a birth or death at time $t > T$. When a point $v$ dies from $D$ at time $t$, it also dies in $X$, and the $X(t) \subseteq D(t)$ property is maintained.

Let $r(x, v) = K^{-1} f(x + v)/f(x)$. If a point $v$ is born at time $t$ into the dominating process, then the point is also born into $X$ when the mark on $v$ satisfies $M(t) \leq r(x, v)$. By only accepting the birth in $D$ as a birth in $X$ with probability $r(x, v)$, the birth rate for $X$ is reduced from $b(x, v)$ to $b(x, v)r(x, v) = f(x+v)/f(x)$. This last ratio is known as the *Papangelou conditional intensity*, and the method of only selecting certain births is known as *thinning* the process (see Appendix G of [22] for a careful construction of the thinning procedure.)

Here $f(x+v)/f(x)$ is the *Papangelou conditional intensity*, and the thinning property of Poisson processes means that this procedure yields a process $X$ with birth rate $b(x, v) = (K)(K^{-1})f(x + v)/f(x) = f(x + v)/f(x)$. Together with the death rate of 1, this gives reversibility as in (9).

Now to add the swap move. For repulsive processes, often $f(x + v)/f(x)$ is small, but there will exist $w \in x$ such that $f(x - w + v)/f(x)$. This is where it is natural to have a swap take place. Suppose there exist probabilities $p(x, w, v)$ such that

$$K^{-1} \frac{f(x + v)}{f(x)} + \sum_{w \in x} p(x, w, v) \leq 1, \tag{11}$$

and

$$f(x)p(x, w, v) = f(x + v - w)p(x + v - w, v, w). \tag{12}$$

Number points in $x$ as $(w_1, w_2, \ldots, w_{\#x})$. Set $\ell_k = [K^{-1} f(x+v)/f(x)] + \sum_{i=1}^{k-1} p(x, w_i, v)$, and $u_k = [K^{-1} f(x + v)/f(x)] + \sum_{i=1}^{k} p(x, w_i, v)$. If $M(v) \leq K^{-1} f(x + v)/f(x)$, then $v$ is added to the configuration, and if $M(v) \in [\ell_k, u_k]$, then $v$ is added and $w_k$ is removed from the configuration.

The probability of swapping from $x$ to $x + w - v$ is just $p(x, w, v)$, while that of swapping from $x + w - v$ to $x$ is $p(x + v - w, v, w)$, and detailed balance for swap moves is maintained. This procedure is now illustrated for two models.

**Strauss model** In the Strauss model ([18],[26]), the density has a factor which is exponential in the number of pairs of points that lie within distance $R$ of each other Let $\rho$ be a metric on $S$, then the density can be written:

$$f_S(x) = Z^{-1}_{(\beta_1,\beta_2,R)} \beta_1^{\#x} \beta_2^{s(x)}, \quad s(x) = \sum_{\{v,v'\}: v \in x, v' \in x \setminus \{v\}} \mathbf{1}(\rho(v, v') \leq R), \tag{13}$$

where $Z_{(\beta_1,\beta_2,R)}$ is the normalizing constant for the density. As noted in [18], in order for $Z_{(\beta_1,\beta_2,R)}$ to be finite (and hence for the density to exist) $\beta_2$ must be at most 1.

Typically $S$ is a bounded subset of $\mathbf{R}^d$ for $d \in \{1, 2, 3\}$, and $\lambda$ is Lebesgue measure. Figure 2 illustrates a random draw from this density for $\lambda$ equal to Lebesgue measure and $S = [0, 1]^2$. This sample was generated using the techniques of the next section.

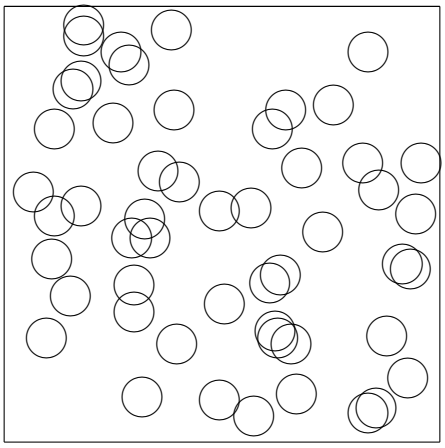


Figure 2: Strauss model with $S = [0,1]^2$, $\beta_1 = 100$, $\beta_2 = 0.50$, $R = 0.1$

Since $\beta_2 \leq 1$, $K = \beta_1$ and births at $v$ in the dominating process are accepted with probability $\beta_2^{n(v,x)}$, where $n(v,x) = \sum_{w \in x} \mathbf{1}(\rho(v,w) \leq R)$ is the number of points currently in the configuration within distance $R$ of the proposed point $v$.

Let Bern($p$) denote a Bernoulli distribution with parameter $p$. One way to generate $B \sim \text{Bern}(\beta_2^{n(v,x)})$ is to generate $B_w \overset{\text{iid}}{\sim} \text{Bern}(\beta_2)$ for each $w$ within distance $R$ of $v$ and let $B = \prod B_w$. Only when every one of the neighbors draws a 1 is the birth accepted.

Now suppose exactly one of the $B_w = 0$ while the rest are 1. This is the situation where a swap move can be used: point $v$ is added to the state while point $w$ is removed. For the Strauss process this makes $p(x,w,v) = \mathbf{1}(\rho(v,w) \leq R)(1-\beta_2)\beta_2^{n(v,x)-1}$, so equations (11) and (12) are satisfied. Therefore this is a legal swap move.

The following pseudocode shows how a birth or death in the dominating chain can be used to make a move in the underlying chain with $f_S$ as the stationary density. In order to run this chain, first run the dominating process chain with $K = \beta_1$. Then take each move ordered by time and apply the following procedure. Here the input variable *move* contains the type of move and the location of the point, and $M$ is the uniform mark attached to a birth event. (Lines 8 and 10 ensure that after testing $M$ against $\beta_2$, the value of $M$ is changed so once again $M$ is uniform over $[0,1]$.)

**Strauss birth death swap chain**
*Input: move*, $M$, $X$, *Output:* $X$

```
1)  If move = death of point w
2)    Let X ← X − w
3)  Else (there was birth at point v)
4)    Let A = {w ∈ X : ρ(w, v) ≤ R}
5)    For each w ∈ A do
6)      If M < β_2
7)        Let A ← A \ w
8)        Let M ← M/β_2
9)      Else
10)       Let M ← (1 − M)/(1 − β_2)
11)   If #A ≤ 1
12)     Let X ← X + v − A
```

**Pairwise interaction point processes** The Strauss process can be generalized to the pairwise interaction point process [18]. The new density is:

$$g(x) = \beta_1^{\#x} \prod_{\{v,v'\} \in x} \phi(\{v, v'\}),$$

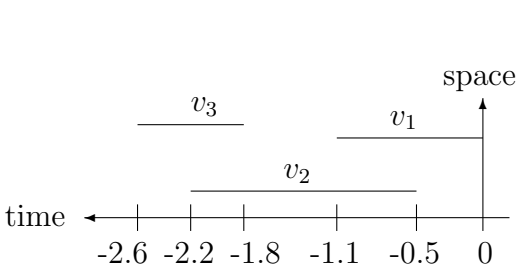
for $\phi$ an arbitrary nonnegative function. For a Strauss process $\phi(\{v, v'\}) = \beta_2^{\mathbf{1}(\rho(v,v') \le R)}$.

When $\phi(\cdot) \le 1$, the same pseudocode as for the Strauss process can be used with a few changes. In line 4 set $A = \{w \in X : \phi(\{v, w\}) > 0\}$, and in lines 6, 8, and 10 change $\beta_2$ to $\phi(v, w)$.

## 4 Perfect simulation by dominated CFTP

The approach used here for perfect simulation is dominated CFTP [20]. The basic idea is to consider a stationary process $\{X(t)\}_{t=-\infty}^{0}$, and try to determine $X(0)$, the configuration of the process at time 0. First note that the dominating process $D$ is time reversible, and so can also be considered over $t \in (-\infty, 0]$. Fix $T > 0$. At this stage, $D(-T)$ is known, but $X(-T)$ is not. Because the $D$ and $X$ processes are coupled together, knowing the births and deaths used to update $D$ for $t \in [-T, 0]$ can sometimes be used to evaluate $X(0)$ without knowing $X(-T)$. In this case, the known value of $X(0)$ is returned by the algorithm. Otherwise, the algorithm considers a larger interval, such as $[-2T, 0]$. First events in $[-2T, -T]$ are generated (the events in $[-T, 0]$ are reused) and again the algorithm tries to find $X(0)$. The process repeats until $X(0)$ is found.

Now for the details. Suppose $X$ is an underlying process with marked dominating process $(D, M)$ as in Section 3. Fix a number of events $N$. Then moving backward in time from time 0, consider the first $N$ births and deaths to occur in the dominating process. Draw $D(0)$ as a spatial Poisson point process with intensity $K\lambda$. Then as the process moves backwards in time, birth and deaths in $D$ are generated starting from




$A$

| time | type | pt. | mark |
|---|---|---|---|
| -0.5 | death | $v_2$ | |
| -1.1 | birth | $v_1$ | .2311 |
| -1.8 | death | $v_3$ | |
| -2.2 | birth | $v_2$ | .4860 |
| -2.6 | birth | $v_3$ | .8913 |

Figure 3: Sample run of Dominating process

this initial configuration. Figure 3 illustrate what a set of moves $A$ looks like after running the process backwards in time.

Now the underlying process can only have births at the same time as births in the dominating process, and therefore the state of the chain at time $-t$ is a collection of points $X(-t) \subseteq D(-t)$ (which is why $D$ is referred to as the dominating process.) As pointed out in [20], it is always possible to create upper and lower configurations $U(t)$ and $L(t)$ such that

$$L(t) \subseteq X(t) \subseteq U(t) \subseteq D(t) \text{ for all } t \in (-\infty, 0]. \tag{14}$$

In [20] this was called the *sandwiching property*. The process $(L(t), U(t))$ can also be thought of as a bounding process (see [14]).

In particular, suppose that $D(t)$ is known backwards in time through the first $N$ events. Call the time where event number $N$ occurred (moving backward in time) $\tau_N$. Then set $L_N(\tau_N)$ to the empty configuration, and $U_N(\tau_N)$ to $D(\tau_N)$. Every time there is an event at time $t$ (either a birth or death in the dominating process) make sure that $U_N$ and $L_N$ continue to bound $X$ once the event updates the chain. Then a simple induction argument shows that $L_N(0) \subseteq X_N(0) \subseteq U_N(0)$. Note if $L_N(0) = U_N(0)$, then $X_N(0)$ is trapped between them and also equals this common value. This is the "coupling" part of CFTP.

The "from the past" part of CFTP works as follows. Suppose that $L_N(0) \neq U_N(0)$. Then increase the value of $N$ and try again. Let $N' > N$. The first $N$ events for the dominating process have already been generated, these same events must be used in subsequent evaluations of the bounding process. Therefore, only $N' - N$ additional events need to be generated. Once these events have been generated, run $L_{N'}$ and $U_{N'}$ forward until $L_{N'}(0)$ and $U_{N'}(0)$ can be compared.

The updates must have the *funneling* property (see [3]): for all $t' > t$ and $N' > N$

$$L_N(t) \subseteq L_{N'}(t) \subseteq U_{N'}(t) \subseteq U_N(t) \Rightarrow L_N(t') \subseteq L_{N'}(t') \subseteq U_{N'}(t') \subseteq U_N(t'). \tag{15}$$

In our approach, $U_N(t) \subseteq D(t)$ for all $t$ and $N$. Moreover, $L_N(\tau_N) = \emptyset$ and $U_N(\tau_N) = D(\tau_N)$. Therefore, for any $N' > N$, $L_N(\tau_N) \subseteq L_{N'}(\tau_N) \subseteq U_{N'}(\tau_N) \subseteq U_N(\tau_N)$. Therefore, if $L_N(0) = U_N(0)$ for some $N$, then $L_{N'}(0) = U_{N'}(0)$ for all $N' > N$ as well, so it is not necessary to try every value of $N$. Propp and Wilson [24] noted the advantages of doubling $N$ at each step, and in the pseudocode below that is how

the algorithm proceeds. The choice of $N_{initial}$ is up to the user, but note that $L_N(0)$ cannot equal $U_N(0)$ unless every point in $D(\tau_N)$ has died by time 0. Therefore, a simple reasonable choice for $N_{initial}$ is the expected number of points in the dominating process at a fixed time, $\lambda(S)$.

**Dominated coupling from the past**
*Input:* $N_{initial}$ *Output:* $X(0)$
1) **Draw** $P \leftarrow \text{Poisson}(K\lambda(S))$, **let** $N' \leftarrow N_{initial}$, **let** $N \leftarrow 0$
2) **Draw** $x$ as $P$ independent draws from $\lambda(\cdot)/\lambda(S)$
3) **Repeat**
4) **Let** $x, A \leftarrow$ **Backward Dominating**$(x, N' - N)$
5) **Let** rows $N+1$ through $N'$ of table $B$ be table $A$
6) **Generate** $(L_{N'}, U_{N'})$ starting from $(\emptyset, x)$ using $B$
7) **Let** $N \leftarrow N'$, **let** $N' \leftarrow 2N$
8) **Until** $L_N(0) = U_N(0)$
9) **Let** $X(0) \leftarrow L_N(0)$

The **Backward Dominating** subroutine generates events backwards in time starting from a given state of the dominating chain. Note that dominated CFTP does not need to know the times of the events, and so unlike the **Dominating Process** code, the **Backward Dominating** procedure does not record times of events.

**Backward Dominating**
*Input:* $x$, $N$ *Output:* $x$, $A$
1) **For** $i$ from 1 to $N$ do
2) **Draw** $t_d \leftarrow \text{Exp}(K\lambda(S))$, **draw** $t_b \leftarrow \text{Exp}(\#x)$, **draw** $M \leftarrow \text{Unif}([0,1])$
3) **If** $t_d < t_b$ (then there is a death)
4) **Draw** $v \leftarrow \lambda(\cdot)/\lambda(S)$, **let** $x \leftarrow x + v$, **add** row ["death" $v$] to $A$
6) **Else** (there is a death)
7) **Draw** $w \leftarrow \text{Unif}(x)$, **let** $x \leftarrow x - w$, **add** row ["birth" $w$ $M$] to $A$

Kendall and Møller showed (Theorem 2.1 of [20]) that as long as the probability that $D(t)$ visits the empty configuration in $[0,t]$ goes to 1 as $t$ goes to infinity, this procedure will terminate in finite time with probability 1. The resulting configuration $L_N(0) = U_N(0)$ is a draw exactly from the target distribution.

Now consider the question: how should $L$ and $U$ be updated given a particular event in the dominating process so the funneling property (15) holds?

### 4.1 Updating the bounding process

Suppose at time $t$ that point $w$ dies in the dominating process $D(t)$. Then as points die in the dominating process, they die in the underlying process $X(t)$ as well, and so having them die in $U_N(t)$ and $L_N(t)$ means they still bound $X(t)$.

Births and swap moves affect the bounding process in the following fashion. Suppose that $v$ is born, and marked with value $M$. Say that $Y \in [L_N(t), U_N(t)]$ if $L_N(t) \subseteq Y \subseteq U_N(t)$. Then if there exists a $Y$ such that $Y \in [L_N(t), U_N(t)]$ and $v$ is added or swapped into $Y$ given $M$, then $v$ is also added to $U_N(t)$. Alternatively, if for all $Y$ such that $Y \in [L_N(t), U_N(t)]$, the point $v$ is added or swapped into $Y$ given $M$, then $v$ is added to $L_N(t)$.

This setup ensures that if $v$ is added to $L_N(t)$, it is also added to $U_N(t)$. For a swap, there are two possibilities.

1. If for all $Y$ such that $Y \in [L_N(t), U_N(t)]$ point $w$ is swapped out of the process given $M(t)$, then it is removed from $U_N(t)$.
2. If there exists a $Y$ such that $Y \in [L_N(t), U_N(t)]$ and point $w$ is swapped out of the process given $M(t)$, then $w$ is removed from $L_N(t)$.

This is the best update possible for $L_N(t)$ and $U_N(t)$ that maintains both the sandwich and funneling properties.

This method also satisfies equation (15). Consider running $L_N, U_N$ and $L_{N'}$, $U_{N'}$ for some $N' > N$. After $N' - N$ events running $(L_{N'}, U_{N'})$ forward in time to time $\tau_N$, they will start using the same birth and death updates as $(L_N, U_N)$. Since $L_N(\tau_N) = \emptyset$ and $U_N(\tau_N) = D(\tau_N)$, it is true that $L_N(\tau_N) \subseteq L_{N'}(\tau_N) \subseteq U_{N'}(\tau_N) \subseteq U_N(\tau_N)$.

In a birth event at time $t$, a point $v$ is added to $U_{N'}(t)$ if there exists $Y \in [L_{N'}(t), U_{N'}(t)]$ that would add the point $v$. The same $Y$ also lies in $[L_N(t), U_N(t)]$, and so will be added to $U_N(t)$ as well. This preserves $U_{N'} \subseteq U_N$. Similarly, points added to $L_{N'}$ will also be added to $L_N$, and dead points are removed from all processes, therefore (15) holds for one event. A simple induction then verifies that it will hold for any finite number of events.

# 5 Simulation of examples

In this section the general methodology of the previous section is applied to the Strauss process from Section 3 on $S = [0, 1]^2$ with $\lambda$ equal to Lebesgue measure. Notice that as $R$ goes to zero, the number of points that fit inside $S$ without overlap goes to infinity, so this setup can be used to understand the Strauss process as the window grows to $\mathbf{R}^2$.

**Strauss process** When a point dies in the dominating process, it dies in any $X \in [L_N, U_N]$ so it can be safely removed from both and maintain the sandwich property. The interesting cases happen when a point is born. The following pseudocode reflects this structure.

**Strauss bounding swap chain**

*Input: move*, $M$, $L$, $U$, $p_{swap}$ *Output:* $L$, $U$

**If** *move* = death of point $w$
  **Let** $L \leftarrow L - w$, **let** $U \leftarrow U - w$
**Else** (there was birth at point $v$)
  **Let** $A_L = \{w \in L : \rho(w, v) \leq R\}$, **let** $A_U = \{w \in U : \rho(w, v) \leq R\}$
  **For** each $w \in A_U$ do
    **If** $M < \beta_2$
      **Let** $A_U \leftarrow A_U \setminus \{w\}$, **let** $A_L \leftarrow A_L \setminus \{w\}$,
      **Let** $M \leftarrow M/\beta_2$
    **Else**
      **Let** $M \leftarrow (1 - M)/(1 - \beta_2)$
  **Let** $[L, U] \leftarrow$ **Birth update**$(v, A_L, A_U, p_{swap})$

To compare the swap chain with the no swap chain, it will be convenient to allow the **Birth update** procedure to not always execute the swap. When it is possible to execute the swap, a Bernoulli with parameter $p_{swap}$ will be rolled. Only if this Bernoulli is 1 will the swap be executed. Therefore this new chain is a mixture of the no swap chain and the swap chain. When $p_{swap} = 0$, it is the original chain with no swap, when $p_{swap} = 1$ it is the new chain that always swaps when possible.

**Birth Update**

*Input:* $v, A_L, A_U, p_{swap}$ *Output:* $L$, $U$

1) **Draw** $S \leftarrow \text{Bern}(p_{swap})$
2) **If** $S = 0$, $\#A_L = 0$, $\#A_U = 0$, **then** $L \leftarrow L \cup \{v\}$, $U \leftarrow U \cup \{v\}$
3) **If** $S = 0$, $\#A_L = 0$, $\#A_U \geq 1$ **then** $U \leftarrow U \cup \{v\}$
4) **If** $S = 1$, $\#A_L = 0$, $\#A_U = 0$, **then** $L \leftarrow L \cup \{v\}$, $U \leftarrow U \cup \{v\}$
5) **If** $S = 1$, $\#A_L = 0$, $\#A_U = 1$, **then** $L \leftarrow L \cup \{v\}$, $U \leftarrow U \cup \{v\} \setminus A_U$
6) **If** $S = 1$, $\#A_L = 0$, $\#A_U > 1$, **then** $U \leftarrow U \cup \{v\}$
7) **If** $S = 1$, $\#A_L = 1$, $\#A_U = 1$, **then** $L \leftarrow L \cup \{v\} \setminus A_L$, $U \leftarrow U \cup \{v\} \setminus A_U$
8) **If** $S = 1$, $\#A_L = 1$, $\#A_U > 1$, **then** $L \leftarrow L \setminus A_L$, $U \leftarrow U \cup \{v\}$

**Theorem 5.1.** *The above update satisfies the sandwich property.*

*Proof.* For deaths, as noted earlier the sandwich property follows immediately. For births, several cases must be considered. First let $S = 0$ (the no swap case).

Line 2: $\#A_L = 0$, $\#A_U = 0$. Then for all $Y \in [L, U]$, $v$ has no neighbors within distance $R$, and so it is added to $Y$. Therefore it should be added to both $L$ and $U$.

Line 3: $\#A_L = 0$, $\#A_U \geq 1$. Then for $Y = L$, the point $v$ will be added, but for $Y = U$, the point $v$ is not. Therefore using the guidelines of Section 4.1, $v$ is added to $U$ but not to $L$.

Other case: $\#A_L > 0$, Then for all $Y \in [L, U]$, the point $v$ will not be born, so both $L$ and $U$ remain unchanged, so no line of the code represents this case.

Now suppose $S = 1$ (so a swap move occurs if possible.)

Line 4: $\#A_L = 0, \#A_U = 0$. In this case, for all $Y \in [L, U]$, the birth is accepted, and so $v$ is added to both $L$ and $U$.

Line 5: $\#A_L = 0, \#A_U = 1$. Let $Y \in [L, U]$, and say $\{w\} = A_U$. If $w \notin Y$ then $v$ is just born. However if $w \in Y$, then $w$ is removed from $Y$ and $v$ is added. Hence $w$ should be removed from $U$ and $v$ added to $L$ and $U$.

Line 6: $\#A_L = 0, \#A_U > 1$. For $Y = L$, $v$ will be born, so must be added to $U$. However, for $Y = U$, $v$ is not added, so $L$ remains unchanged.

Line 7: $\#A_L = 1$, $\#A_U = 1$. This means $A_L = A_U$, so for all $Y \in [L, U]$, the single point in $A_L$ is removed and $v$ is added. Hence this is done for both $L$ and $U$.

Line 8: $\#A_L = 1$, $\#A_U > 1$. For $Y = L$, the point $v$ is added and the point in $A_L$ is removed. So $v$ must be added to $U$ and $A_L$ removed from $L$. For $Y = U$, the point $v$ is neither added nor is $A_L$ removed, so $v$ is not added to $L$ and $A_L$ is not removed from $U$.

Other case: $\#A_L > 1$. Then for all $Y \in [L, U]$ no swap or birth occurs, so both $L$ and $U$ remain unchanged, and no line is needed.

□

Figure 4 shows the advantage gained by using the swap move. The times are measured in number of events generated by the algorithm in order to be comparable with the times needed for a regular Markov chain approach. On the left are the raw number of times for the chain with no swap ($p_{swap} = 0$) and with the swap ($p_{swap} = 1$). The plot on the right shows the ratio of these two times. Note that as $\beta_1$ gets larger, the speedup gained by using the swap move also increases.

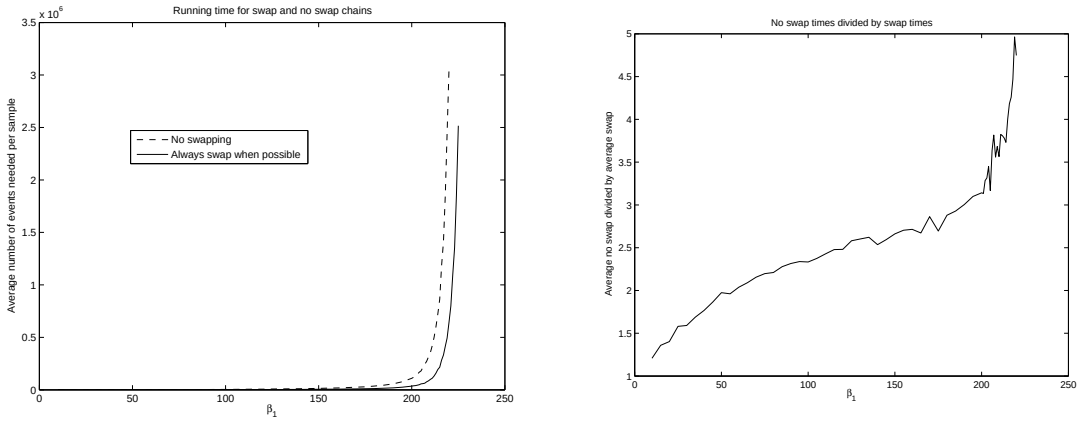



Figure 4: Running time of dCFTP for Strauss model on $S = [0,1]^2$, $\beta_2 = .5$, $R = .1$.

## 6 Running time

Now consider how many events must be generated before the dominated coupling from the past procedure terminates. Deaths in $U_N(t) - L_N(t)$ cause the bounding process to move together, while births can add a point to $U_N(t)$ but not to $L_N(t)$. Therefore it is reasonable that the perfect simulation algorithm will run faster in situations where the birth rate is low. If $N$ events were generated backwards in time, let $U_N$ and $L_N$ denote the corresponding bounding process. Recall that if $U_N(0) = L_N(0)$, then $U_{N'}(0) = L_{N'}(0)$ for all $N' \geq N$.

To develop the best theoretical bound on the mixing time of the Markov chain, it turns out (as in [7]) to be useful not to always execute the swap move. Suppose that when the possibility of a swap exists, the swap move is only executed with probability 1/4. This can be viewed as a mixture of the original and swap move chains.

**Theorem 6.1.** *Suppose that $N$ events are generated backwards in time and then run forward to get $U_N(0)$ and $L_N(0)$. Let $B(v,R)$ denote the area within distance $R$ of $v \in S$, let $r = \sup_{v \in S} \lambda(B(v,R))$.*

*If $\beta_1(1-\beta_2)r < 1$, then for the chain without the swap move*

$$\mathbf{P}(U_N(0) \neq L_N(0)) \leq 2\exp(-.09N) + \beta_1\lambda(S)\exp(-N(1-\beta_1(1-\beta_2)r)/(4\beta_1\lambda(S)). \tag{16}$$

*If $\beta_1(1-\beta_2)r < 2$, then for the chain where a swap is executed with probability $1/4$,*

$$\mathbf{P}(U_N(0) \neq L_N(0)) \leq 2\exp(-.09N) + \beta_1\lambda(S)\exp(-N(1-.5\beta_1(1-\beta_2)r)/(4\beta_1\lambda(S)). \tag{17}$$

Without a swap move, it is possible to prove that the dominated CFTP algorithm runs in polynomial time when $\beta_1(1-\beta_2)r < 1$. Use of a swap move with $1/4$ probability yields a better analysis: for fixed $\beta_2$ and $r$, the algorithm is guaranteed to run in polynomial time over twice the range of $\beta_1$ as before, for $\beta_1(1-\beta_2)r < 2$. When dominated CFTP runs in polynomial time, the underlying Markov chain is also rapidly mixing.

Why the value of $1/4$ for the probability? This is an artifact of the proof technique. The theorem only gives sufficient, not necessary conditions for the algorithm to be fast, and simulation experiments indicate that the algorithm actually takes the fewest steps when the swap moves are used as often as possible (reasons why this is true are noted in the proof of the theorem.)

Theorem 6.1 has immediate consequences for the expected running time of dominated CFTP. Recall that in dCFTP the number of events was doubled each time. Say $\mathbf{P}(U_N(0) \neq L_N(0)) \leq a\exp(-bN)$, and let $T$ be the number of events generated in a call of dCFTP. Then for $T \geq t$, dCFTP must have failed on a run of length at least $t/2$. So

$$\mathbf{E}[T] = \sum_{N=1}^{\infty} \mathbf{P}(T \geq N) \leq \left[\sum_{N=1}^{\lceil (2/b)\ln a\rceil} 1\right] + \sum_{N=\lceil (2/b)\ln a\rceil}^{\infty} a\exp(-bN/2), \tag{18}$$

which makes $\mathbf{E}[T] = O(\ln a/b)$, and the mean running time $O(\beta_1\lambda(S)(\ln \beta_1\lambda(S)))$ for the no swap chain when $\beta_1(1-\beta_2)r < 1$ and in the $1/4$-swap chain when $\beta_1(1-\beta_2)r < 2$.

*Proof of Theorem 6.1.* As in the description of dCFTP in Section 4, the algorithm begins by generating a Poisson spatial point process with parameter $\beta_1\lambda(S)$ at time 0, generate $N$ birth-death events backwards in time. Then $U_N$ becomes this state, $L_N$ is the empty configuration, and the bounding processes are run forward in time. Let $Q(t) = U_N(t) \setminus L_N(t)$. Then the chains have come together if and only if $\#Q(0) = 0$.

**Strauss no swap move** Here $p_{swap} = 0$. All individual death rates are 1, so the rate of deaths of points in $Q(t)$ is just $\#Q(t)$. Each death reduces $Q(t)$ in size by 1. Call this a good event. So the rate of good events is $\#Q(t)$.

For $\#Q(t)$ to increase by 1 (call this a bad event), a birth must occur at $v$ and be added to $U_N(t)$ but not $L_N(t)$. Let $w$ be any point in $Q(t)$. Then for $Q(t)$ to give rise to another point in $Q(t)$, a point $v$ must be born within distance $R$ of $w$ and the Bern($\beta_2$) roll must be 0. The area surrounding $w$ is at most $r$, and the Bernoulli roll acts as a thinning procedure in a Poisson process. Therefore the rate at which $w$ is creating new points in $Q(t)$ is at most $\beta_1(1-\beta_2)r$, and the overall rate of bad events is at most $\beta_1(1-\beta_2)r\#Q(t)$.

Suppose the rate of bad events is smaller than the rate of good events. The probability that one event occurs in the time interval from $t$ to $t+h$ is proportional to $h$,

the probability that $n$ events occurs is $O(h^n)$. Hence

$$\mathbf{E}[\mathbf{E}[\#Q(t+h)|U(t),L(t)] - \#Q(t)] \leq \mathbf{E}[(\#Q(t)\beta_1(1-\beta_2)r - \#Q(t))h + \sum_{i=2}^{\infty} iO(h^i)].$$

which means

$$\lim_{h\to 0} \frac{\mathbf{E}[\mathbf{E}[\#Q(t+h)|U(t),L(t)] - \#Q(t)]}{h} \leq -\mathbf{E}[\#Q(t)(1-\beta_1(1-\beta_2)r)].$$

Let $q(t) = \mathbf{E}[\#Q(t)]$, and let $\tau_N$ be the time of the $N$th event moving backwards in time. Then $q(\tau_N) \leq \mathbf{E}[\#D(\tau_N)] = \beta_1\lambda(S)$, so together with $q'(t) \leq -q(t)(1-\beta_1(1-\beta_2)r)$:

$$q(t) \leq \beta_1\lambda(S)\exp(-t(1-\beta_1(1-\beta_2)r)).$$

By Markov's inequality, $\mathbf{P}(Q(0) \neq \emptyset) = \mathbf{P}(\#Q(0) \geq 1) \leq q(0)$.

Now fix $N$, the number of events to run back in time, and set $t = N/[4\beta_1\lambda(S)]$. The probability that $Q(0)$ does not equal 0 starting at $-t$ is at most $\exp(-N/[4\beta_1\lambda(S)](1-\beta_1(1-\beta_2)r))$.

Using Chernoff bounds [5], it can be shown that for $A \sim \text{Pois}(\alpha)$, $\mathbf{P}(A > 2\alpha) \leq \exp(-\alpha(2\ln 2 - 2 + 1))$. So after $t$ time, the probability that more than $N/2$ events were generated in a Poisson process with rate $\beta_1\lambda(S)$ is at most $\exp(-(N/4)(2\ln 2 - 2 + 1))$. Both the times of the births and times of deaths (viewed individually) are Poisson processes with rate $\beta_1\lambda(S)$, therefore the probability that either uses more than $N/2$ events (by the union bound) is at most $2\exp(-.09N)$. But if at this time each process used at most $N/2$ events, then moving back in time $N$ events puts the user even farther back in time, and if coalescence occurs at $-t$, it will also occur starting at $\tau_N$. Again using the union bound, the probability of failure is at most

$$2\exp(-.09N) + \exp(-N(1-\beta_1(1-\beta_2)r)).$$

**Strauss with swap move** Now consider what happens when $p_{swap} > 0$. The rate of good events (deaths) remains unchanged, but the rate of bad events changes. Going back to **Birth Update**, lines 2, 4, and 7 leave $\#Q(t)$ unchanged, lines 3 and 6 increase $\#Q(t)$ by 1, line 5 decreases $\#Q(t)$ by 1, and line 8 increases $\#Q(t)$ by 2. Let $b_3$ denote the area of the region where a birth triggers line 3, with $b_4, b_5, b_6$, and $b_8$ defined similarly. Line 3 is only activated when a swap does not occur, and lines 4 through 8 only occur when a swap does occur. Putting all this together, the rate of change from births is:

$$(1-p_{swap})b_3 + p_{swap}b_5(-1) + p_{swap}b_6 + p_{swap}b_8(2).$$

Note that $b_3 = b_5 + b_6$ since the conditions on lines 5 and lines 6, neglecting the condition on $S$, form a partition of the conditions in line 3. So the rate can be simplified to

$$(1-2p_{swap})b_5 + b_6 + 2p_{swap}b_8.$$

Now, in line 6, $\#A_U - \#A_L \geq 2$, so any point triggering these lines must be within distance $R$ of at least two points in $Q(t)$. Points in $b_5$ or $b_8$ must be within distance

$R$ of at least one point in $Q(t)$. Since each point in $Q(t)$ is adjacent to area at most $r$, this means $b_5 + 2b_6 + b_8 \leq \#Q(t)r$.

The variable $p_{swap}$ can be set to any number from 0 to 1, but setting it to $p_{swap} = 1/4$ gives an upper bound on the bad event rate of $(1/2)b_5 + b_6 + (1/2)b_8 \leq (1/2)\#Q(t)r$. Note that if $b_5 > b_8$, then it makes sense to make $p_{swap} = 1$ to drive the bad event rate as small as possible. While simulations indicate that $p_{swap} = 1$ is the best choice in practice, setting $p_{swap} = 1/4$ gives the best analysis of this algorithm.

Recall the bad event rate when $p_{swap} = 0$ was bounded above by $\#Q(t)r$. With $p_{swap} = 1/4$, the bad event rate is bounded above by $\#Q(t)r/2$, and this factor of two carries throughout the remainder of the proof to give (17). □

Häggström and Steif gave a result similar to the previous theorem for finitary codings for high noise Markov random fields [11], but their analysis involves moving backwards rather than forwards in time, and they did not employ the swap move.

Figure 5 illustrates the mean run time for a fixed value of $\lambda$ as the probability of a swap varies from $p = 0$ up to $p = 1$. The running time (as measured by generated iterations) decreases as the chance of swapping increases. This same phenomenon was noted for the hard-core gas model on graphs in [15].

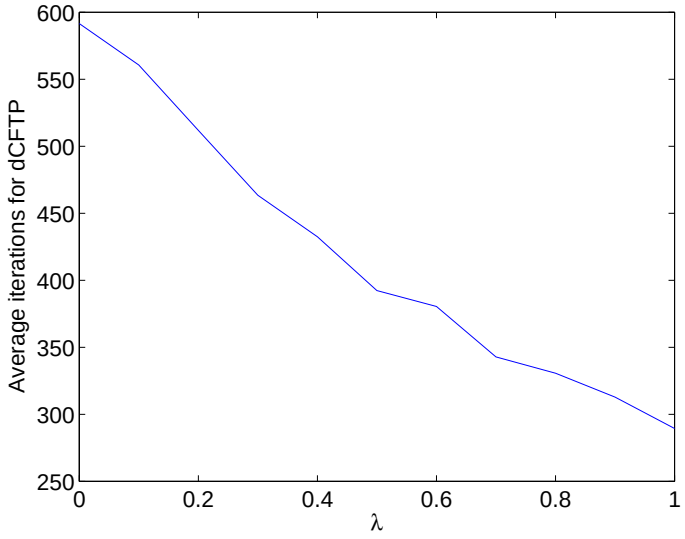



Figure 5: Expected iterations for $\lambda = 40$ with varying probability of swapping

## 7 Conclusions

The regular birth death chains only move when no point blocks the birth of a point in the dominating process. The birth death swap chains move when at most one point blocks the birth of a point in the dominating process. This alone means that more moves are being taken, and helps to explain the improved analysis and improved performance when used for perfect sampling with dominated coupling from the past.

## References


[1] J. Azéma, M. Kaplan-Duflo, and D. Revuz. Measure invariante sur les classes recurrents des processus de Markov. *Z.W.*, 8:157–181, 1967.

[2] A. J. Baddeley and M. N. M. van Lieshout. Area-interaction point processes. *Ann. Inst. Statist. Math.*, 47:601–619, 1995.

[3] K. K. Berthelsen and J. Møller. A primer of perfect simulation for spatial point processes. *Bull. Braz. Math. Soc.*, 33(3):351–367, 2002.

[4] D.S. Carter and P.M. Prenter. Exponential spaces and counting processes. *Z. Wahrsheinlichkeitsth*, 21:1–19, 1972.

[5] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Ann. of Math. Stat.*, 23:493–509, 1952.

[6] P. Clifford and G. Nicholls. Comparison of birth-and-death and Metropolis-Hastings Markov chain Monte Carlo for the Strauss process, 1994.

[7] M. Dyer and C. Greenhill. On markov chains for independent sets. *J. Algorithms*, 35(1):17–49, 2000.

[8] W. Feller. *An Introduction to Probability Theory and its Applications, Volume II.* Wiley, 1966.

[9] G. S. Fishman. *Monte Carlo: concepts, algorithms, and applications.* Springer-Verlag, 1996.

[10] C. Geyer. *Likelihood inference for spatial point processes*, pages 79–140. Chapman & Hall/CRC, 1999.

[11] O. Häggström and J. E. Steif. Propp-Wilson algorithms and finitary codings for high noise Markov random fields. *Combin. Probab. Computing*, 9:425–439, 2000.

[12] O. Häggström, M.N.M. van Leishout, and J. Møller. Characterisation results and Markov chain Monte Carlo algorithms including exact simulation for some spatial point processes. *Bernoulli*, 5:641–658, 1999.

[13] R. D. Harkness and V. Isham. A bivariate spatial point pattern of ants' nests. *Appl. Stat.*, 32:293–303, 1983.

[14] M. Huber. Perfect sampling using bounding chains. *Annals of Applied Probability*, 14(2):734–753, 2004.

[15] M. L. Huber. A faster method for sampling independent sets. In *Proc. 11th ACM-SIAM Sympos. on Discrete Algorithms*, pages 625–626, 2000.

[16] J. Illian, A. Penttinen, H. Stoyan, and D. Stoyan. *Statistical Analysis and Modelling of Spatial Point Patterns.* John Wiley and Sons, Chichester, 2008.

[17] H. Kaspi and A. Mandelbaum. On Harris recurrence in continuous time. *Math. of Oper. Res.*, 19, 1994.

[18] F.P. Kelly and B. D. Ripley. A note on Srauss's model for clustering. *Biometrika*, 63(2), 1976.

[19] W. S. Kendall. Perfect simulation for the area-interaction point process. In *Proceedings of the Sympos. on Probability Towards the Year 2000*, 1995.

[20] W.S. Kendall and J. Møller. Perfect simulation using dominating processes on ordered spaces, with application to locally stable point processes. *Adv. Appl. Prob.*, 32:844–865, 2000.

[21] W. Klein. Potts-model formulation of continuum percolation. *Phys. Rev. B*, 26:2677–2678, 1982.

[22] J. Møller and R. P. Waagepetersen. *Statistical Inference and Simulation for Spatial Point Processes*. Chapman & Hall/CRC, 2004.

[23] C.J. Preston. Spatial birth-and-death processes. *Bull. Inst. Int. Stat.*, 46(2):371–391, 1977.

[24] J. G. Propp and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures Algorithms*, 9(1–2):223–252, 1996.

[25] B. D. Ripley. Modelling spatial patterns (with discussion). *J. Roy. Statist. Soc. Ser. B*, 39:172–212, 1977.

[26] D. J. Strauss. A model for clustering. *Biometrika*, 63:467–475, 1975.

[27] B. Widom and J.S. Rowlinson. A new model for the study of liquid-vapor phase transition. *J. Chem. Phys.*, 52:1670–1684, 1970.